Filtering Tools

Routing Protocol Tools and Route Manipulation

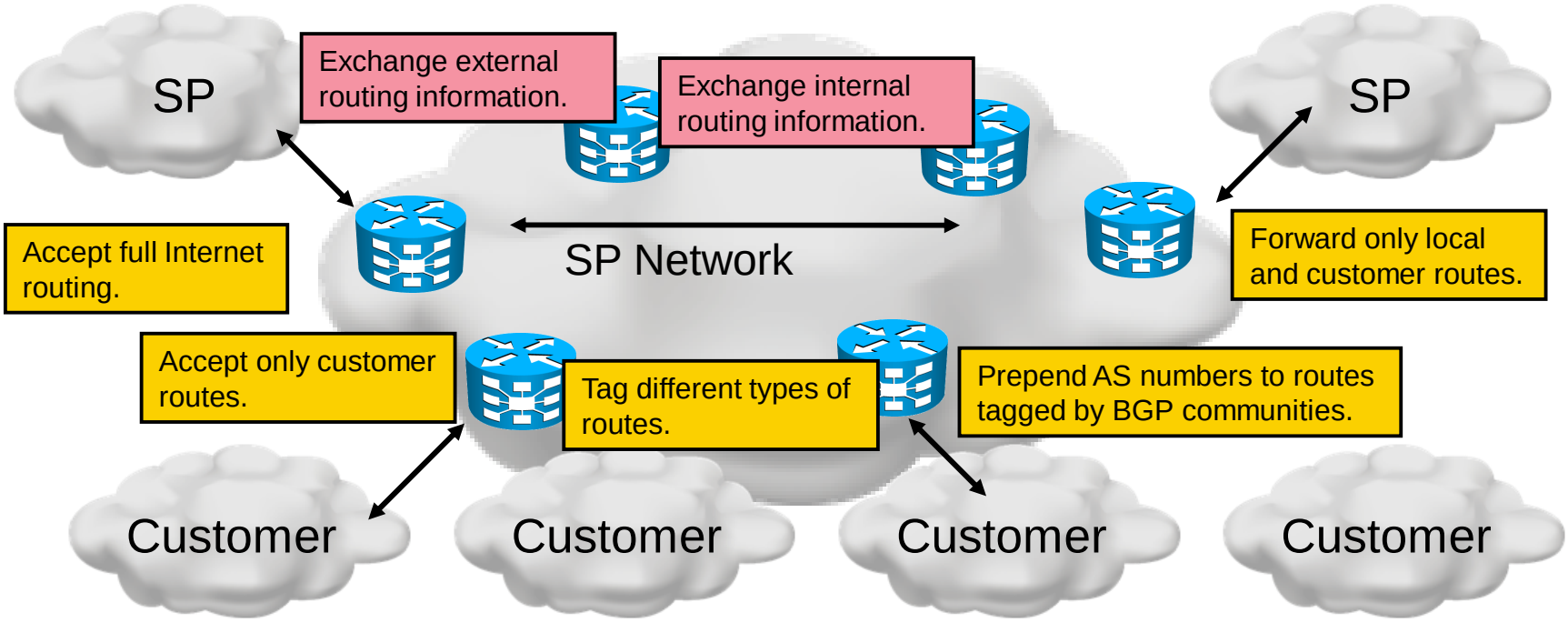
Routing Protocol Tools Overview

Primary objectives:

- Exchange internal routing information
- Exchange external routing information

Secondary high-level objectives:

- Filtering routing updates
- Routing policy implementation (influencing route selection)



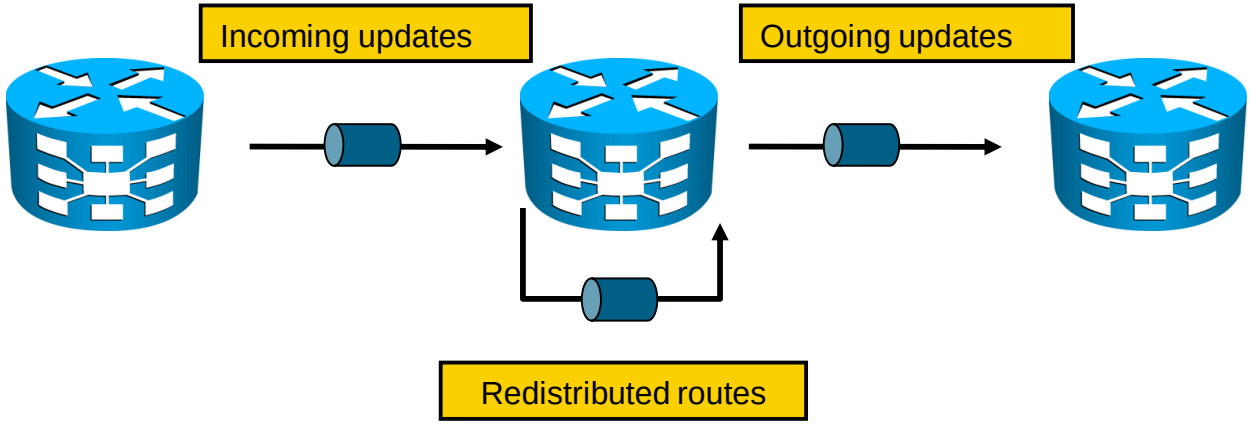
Typical Filtering Objectives

Filter:

- Incoming updates
- Outgoing updates
- Redistributed routes from other routing protocols

Filter based on:

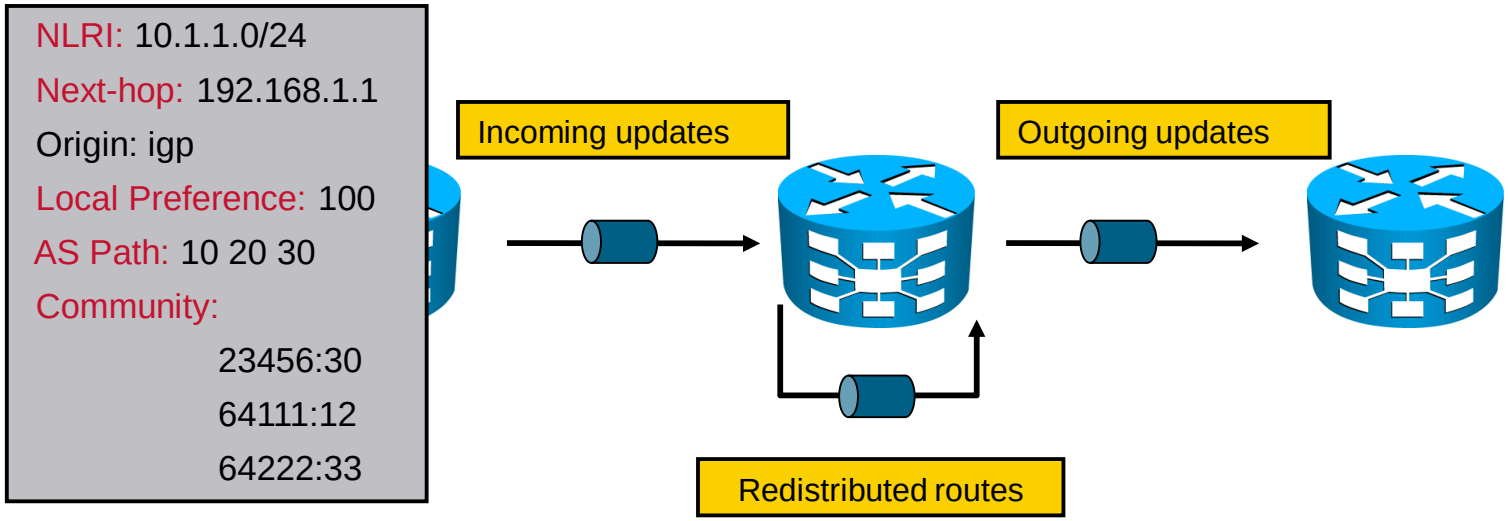
- Prefix and prefix length (subnet mask)
- Update parameters (routing protocol-specific)



Example: Typical BGP Filtering Objectives

Filter BGP based on:

- Prefix and prefix length (subnet mask)
- Next-hop address
- Route source address
- AS path attribute
- BGP community and BGP extended community attributes
- Local preference attribute



Filtering Tools

Prefix lists:

- Used for prefix-based filtering or matching of routes
- Can be used to match on the prefix, route source, or next-hop address

AS path access lists:

- Used in BGP for filtering or route matching based on BGP AS Path attribute

Route maps:

- Primarily used to implement complex routing policies
- Can also be used as a powerful filtering tool

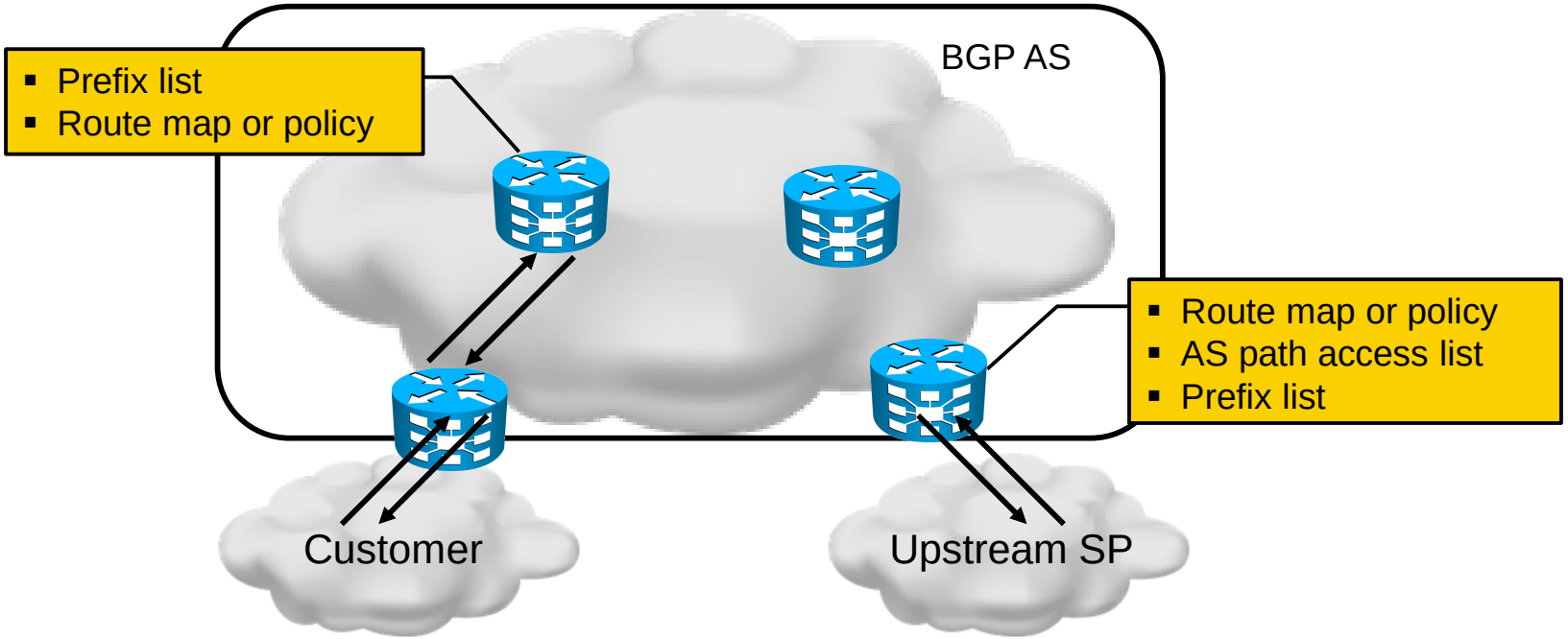
Typical Filtering Objectives in BGP

Typical inbound filtering requirements:

- Permit only customer routes.
- Permit a specific list of routes from peering service providers.

Typical outbound filtering requirements:

- Permit only the default route.
- Permit default route and local routes.
- Permit all routes.



Typical Routing Objectives

- Complex routing policies are most often implemented using BGP.
- Influencing route selection for:
  - Outgoing traffic
  - Incoming traffic
- Routing decision influenced:
  - Locally
  - Remotely (e.g. by customer or downstream service provider)

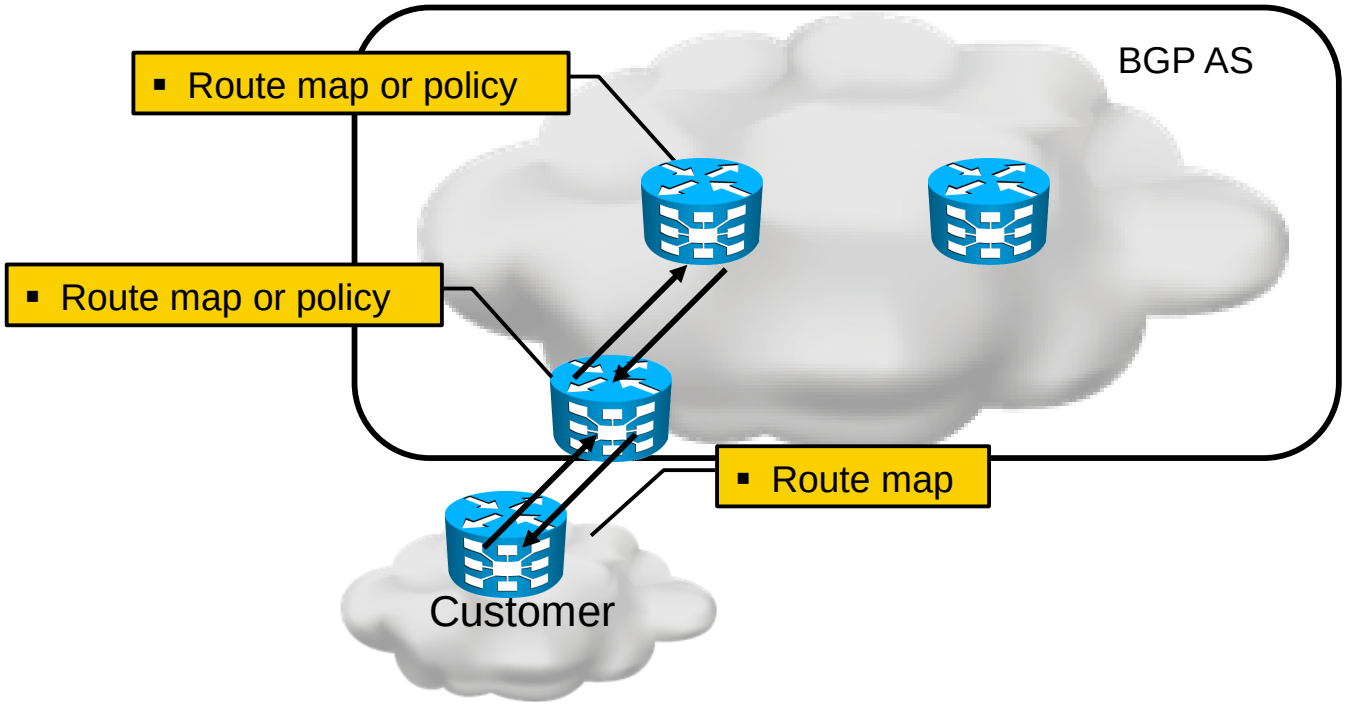
Typical Routing Objectives in BGP

Customer selecting primary or backup ISP:

- AS path prepending by customer
- BGP community sent by customer
- MED

Policy implemented by service provider:

- Setting local preference
- Translating BGP community to local preference



Prefix List Overview

- Designed for route filtering/matching
- Replaces access-lists that were designed for packet filtering/matching

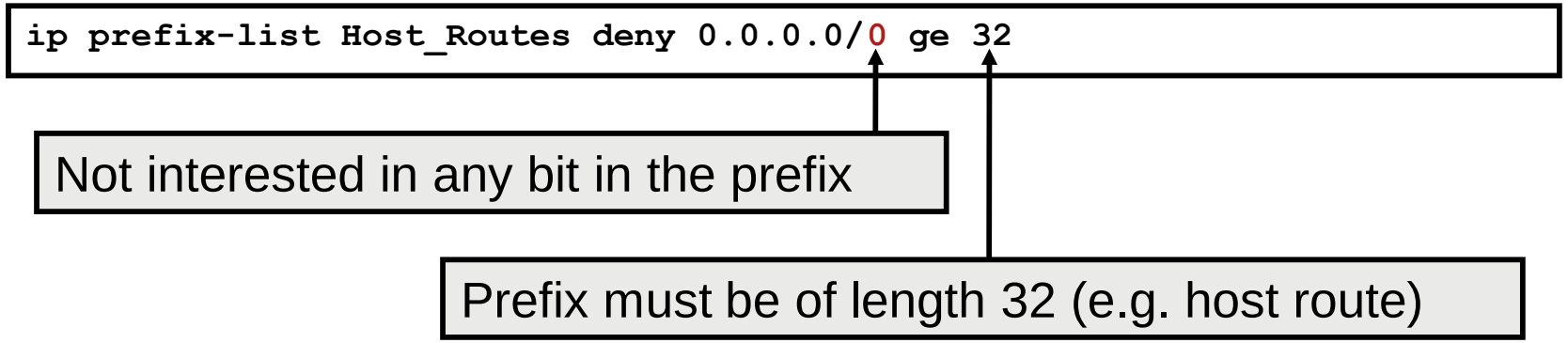
Prefix Lists Syntax

- Each prefix list is identified using a case-sensitive *name*.
- Each prefix list can have one or more lines.
- Edit and order prefix list entries by using line numbers.
- The *network/length* pair identifies the bits in prefixes to match.
- The **ge** and **le** operators identify the length of prefixes to match:
  - **le** :“less or equal” matches any prefix that is shorter or equal in length to the specified value.
  - **ge** :“greater or equal” matches any prefix that is longer or equal in length to the specified value.
  - **ge** *x* **le** *x*

```
Router(config)#
ip prefix-list name [seq num] {deny|permit} net/length [ge len] [le len]
```

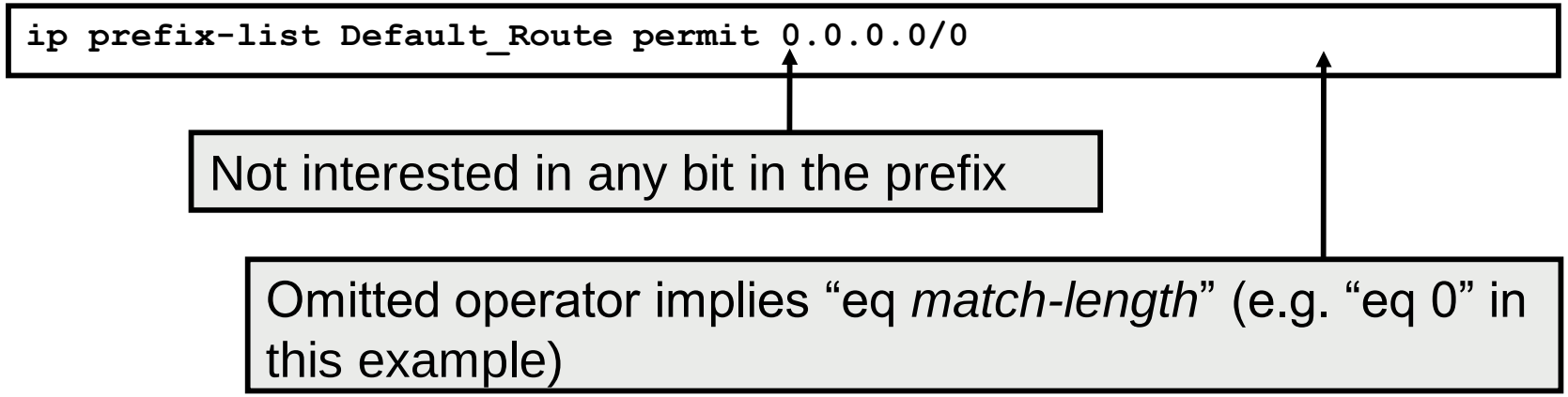
Example: Match Any Host Route

- Host routes are often filtered out to minimize the size of the routing table.



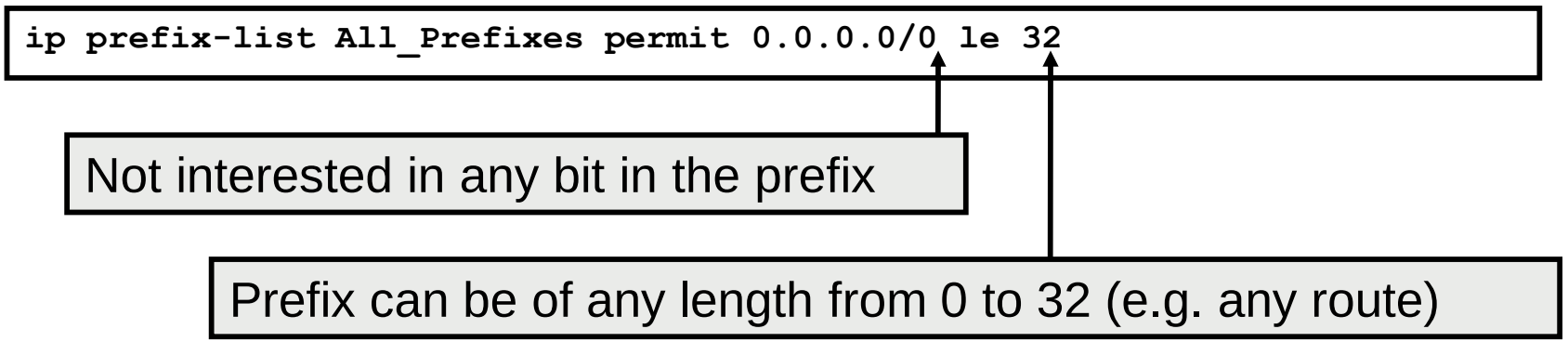
Example: Match Default Route

- Single-homed customers running BGP or multi-homed customers that do not require full Internet routing should receive only the default route.



Example: Match All Routes

- There is no keyword **any** as in access lists.
- Use this example instead, to match any route.



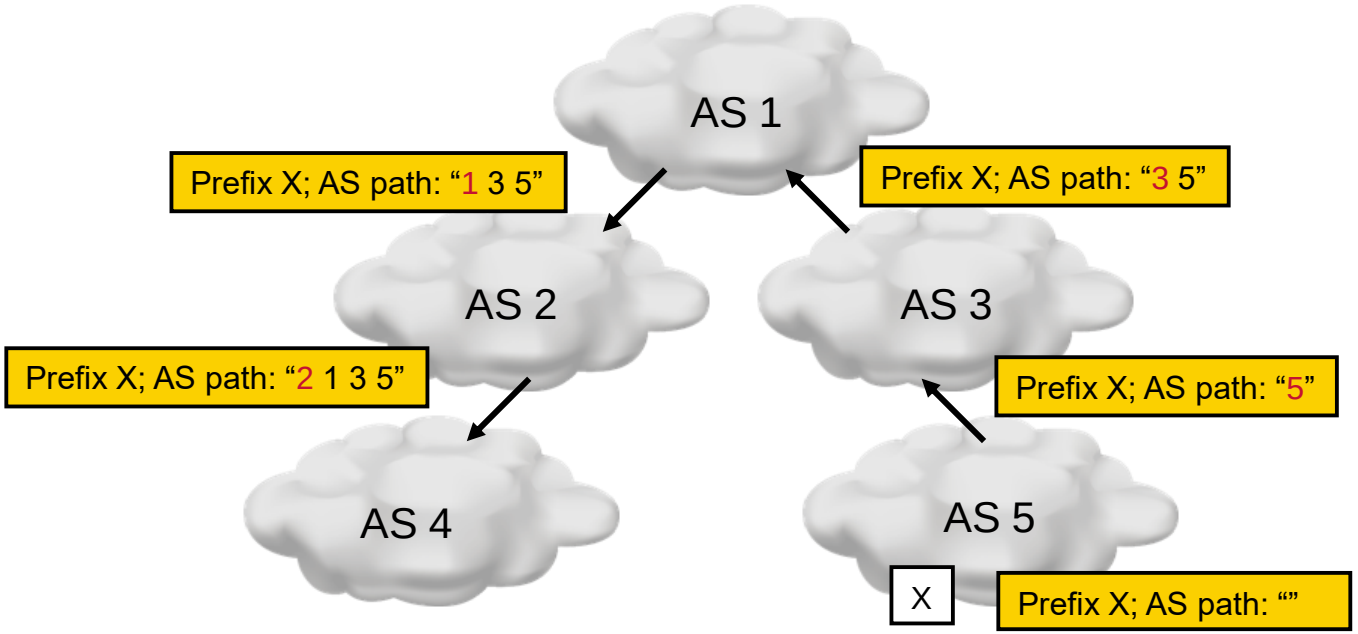
Example: Customer´s Routes

- Match a customer networks.

```
ip prefix-list Customer1_Prefixes seq 10 permit 193.136.194.0/23
ip prefix-list Customer1_Prefixes seq 20 permit 193.136.231.0/24
ip prefix-list Customer1_Prefixes seq 30 permit 193.136.252.0/24
ip prefix-list Customer1_Prefixes seq 40 permit 193.137.101.0/24
ip prefix-list Customer1_Prefixes seq 50 permit 193.137.106.0/23
ip prefix-list Customer1_Prefixes seq 60 permit 193.137.108.0/23
ip prefix-list Customer1_Prefixes seq 70 permit 194.210.181.0/24
ip prefix-list Customer1_Prefixes seq 80 permit 194.210.182.0/23
ip prefix-list Customer1_Prefixes seq 90 permit 194.210.88.0/21
ip prefix-list Customer1_Prefixes seq 100 permit 194.210.104.0/22
ip prefix-list Customer1_Prefixes seq 110 permit 194.210.108.0/23
```

AS Path Based Filtering Overview

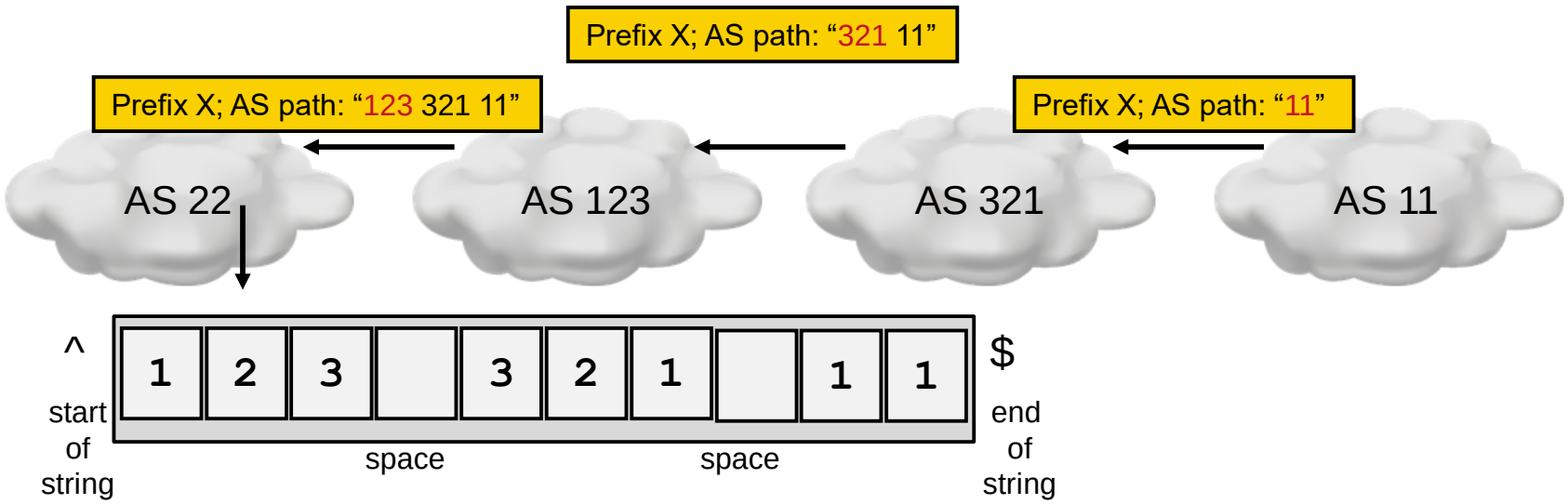
- BGP uses autonomous systems to identify the origin and path of a prefix.
- Each path is identified using a sequence of AS numbers.
- AS path attribute is used to carry the AS path in BGP updates.
- Each egress BGP router prepends its own AS number to the AS path attribute.
- AS path access lists are used to match prefixes based on AS path characteristics.



AS Path Access List Syntax

- Each AS path access list is identified using a unique number.
- Regular expressions are used to match prefixes based on the contents of the AS path attribute.
- The AS path is processed as a string of characters.

```
Router(config)#
ip as-path access-list acl-number {permit | deny} regexp
```



Regular Expressions, Special Characters

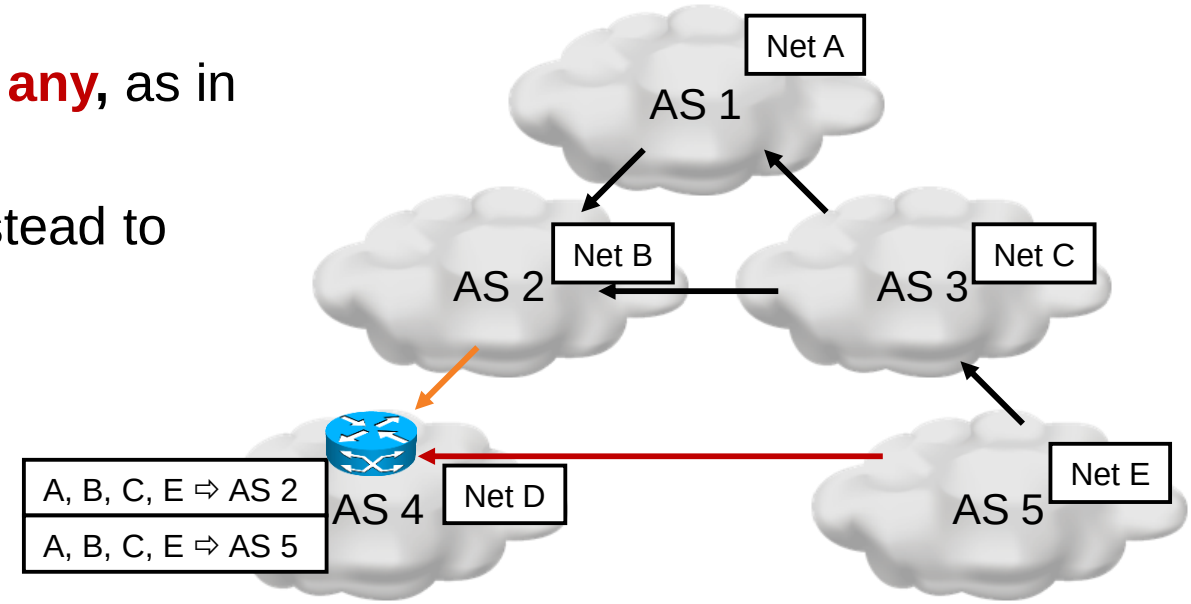
Character	Description
^	matches the start of AS path (e.g. “^20_”)
$	matches the end of AS path (e.g. “^20$”)
_	matches any delimiter (start, end, or space; e.g. “_20_”)
.	matches any single character
*	matches preceding character any number of times including zero (e.g. “.*” “^20(_20)*$”)
+	matches preceding character once or more times (e.g. “^[0-9]+$”)
?	matches preceding character zero or one time (e.g. “^20(_20)?$”)

Commonly Used Regular Expressions

Regular Expression	Description
^$	matches locally originated prefixes
^number$	matches prefixes originating in the specified neighboring AS
_number$	matches prefixes originating in the specified AS
^number_	matches prefixes learned through the specified neighboring AS
^([0-9]+)(_\1)*$	matches prefixes originating in any neighboring AS and allowing prepending
.*	matches all prefixes (e.g. “any”)
.	matches nonlocal prefixes (e.g. all except empty AS path)

Example: Permit All Routes

- There is no keyword **any**, as in access lists.
- Use this example instead to match any route:
- Example:
  - Matches any prefix from any neighbor

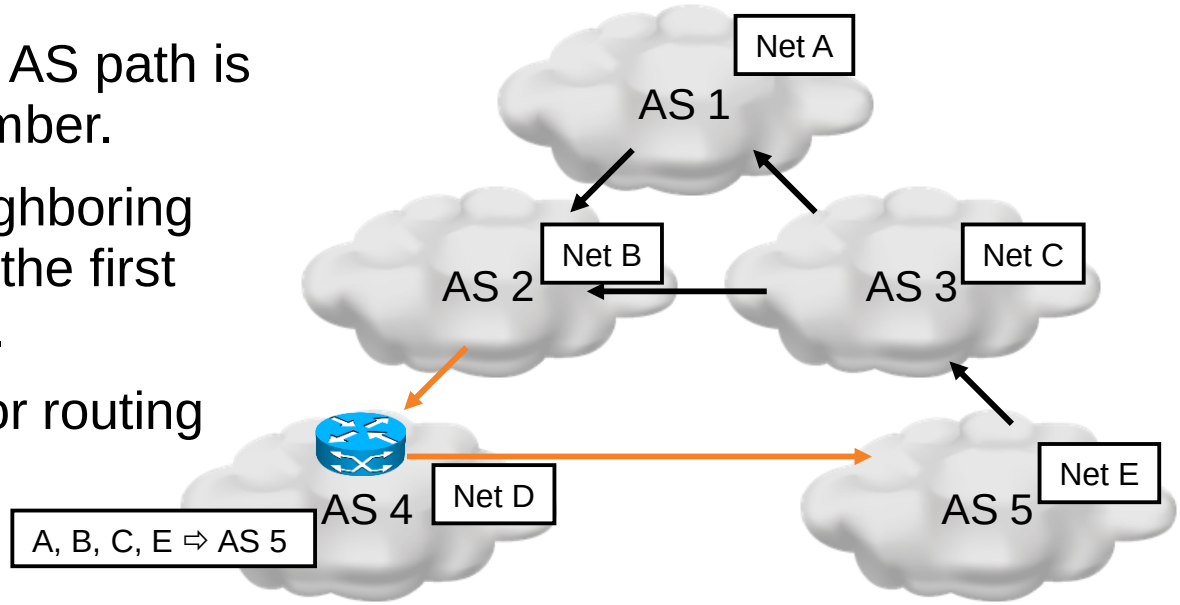



```
ip as-path access-list permit .*
```

Matches any character (.) any number of times (*)

Example: Permit Routes From a Neighbor

- The first number in the AS path is the last prepended number.
- Directly connected neighboring AS is always found as the first number in the AS path.
- Typically this is used for routing policies.
- Example:
  - AS 4 matches any prefix from neighboring AS 5.

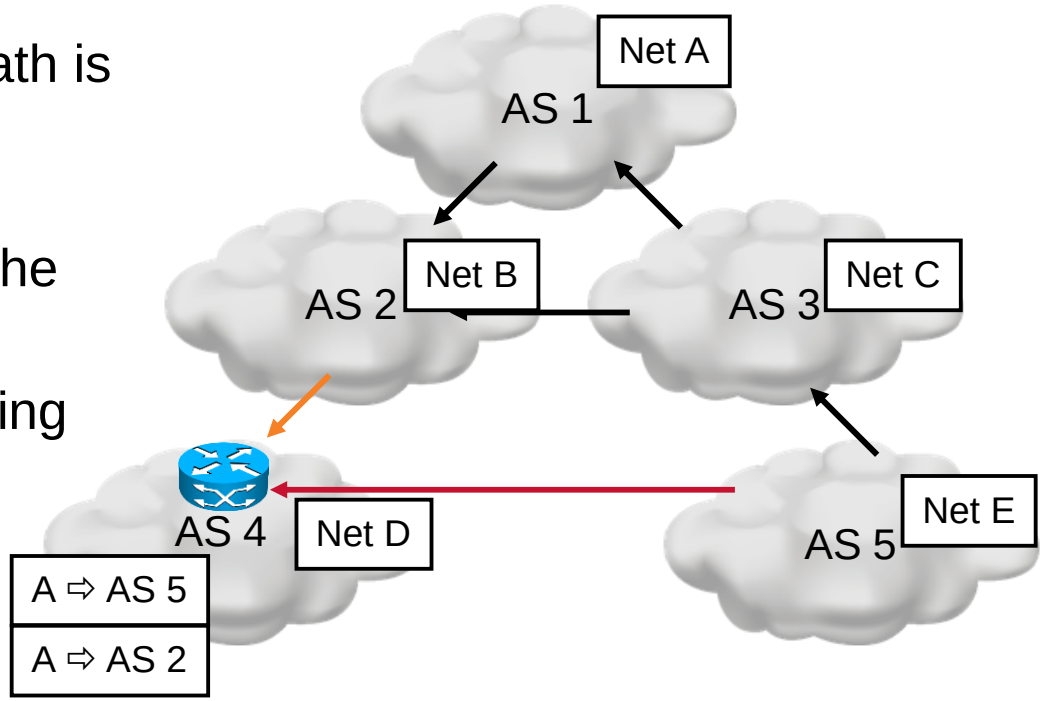



```
ip as-path access-list permit ^5_
```

Matches routes coming from a specific neighbor

Example: Permit Routes Originating in a Specific AS

- The last number in the AS path is the first prepended number.
- The originating AS is always found as the last number in the AS path.
- Typically this is used for routing policies.
- Example:
  - AS 4 matches prefixes originating in AS 1 from any neighboring AS.

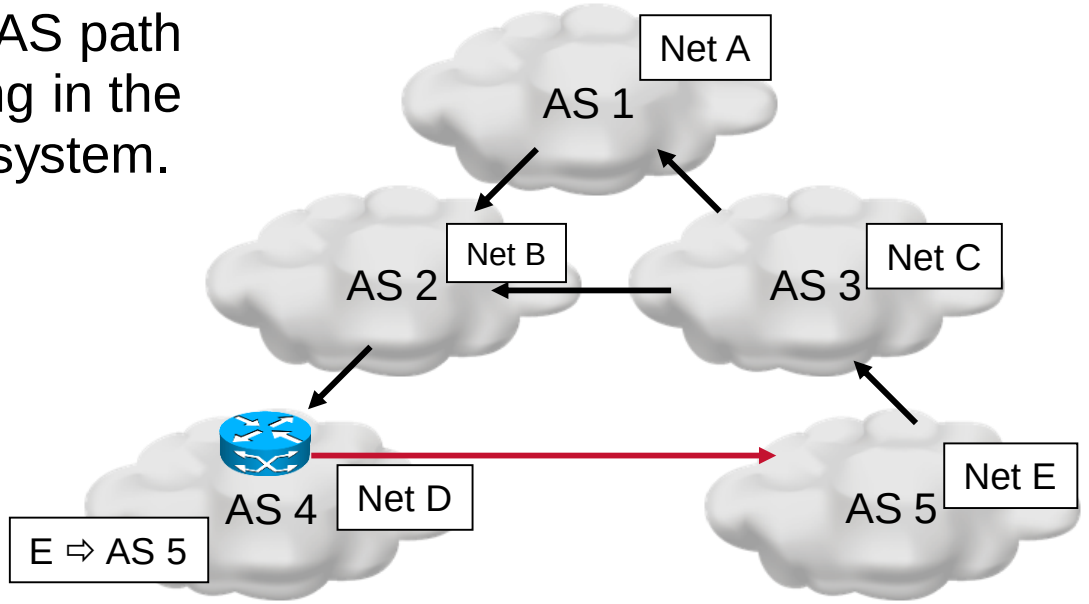



```
ip as-path access-list permit _1$
```

Matches routes coming from a specific AS

Example: Permit Neighboring Local Routes

- A single AS number in an AS path denotes prefixes originating in the neighboring autonomous system.




```
ip as-path access-list permit ^5$
```

Matches a single AS number in the AS path (e.g. prefix originating in a neighboring AS)

Route Maps Overview

- Route maps are a simple language to support complex routing policies, in addition to filtering.
- Route maps are uniquely identified by a case-sensitive name.
- Each route map can have one or more ordered statements identified using the sequence number.
- Each statement can filter updates using permit or deny options.
- Each statement contains zero or more match commands.
- Each statement contains zero or more set commands used to modify routing updates.
- Each statement processes updates matched by the match command and set/modify parameters.

```
Router(config)#
route-map map-tag [permit | deny] [sequence-number]
 match condition
  set parameter value
!
route-map map-tag [permit | deny] [sequence-number]
 match condition
  set parameter value
```

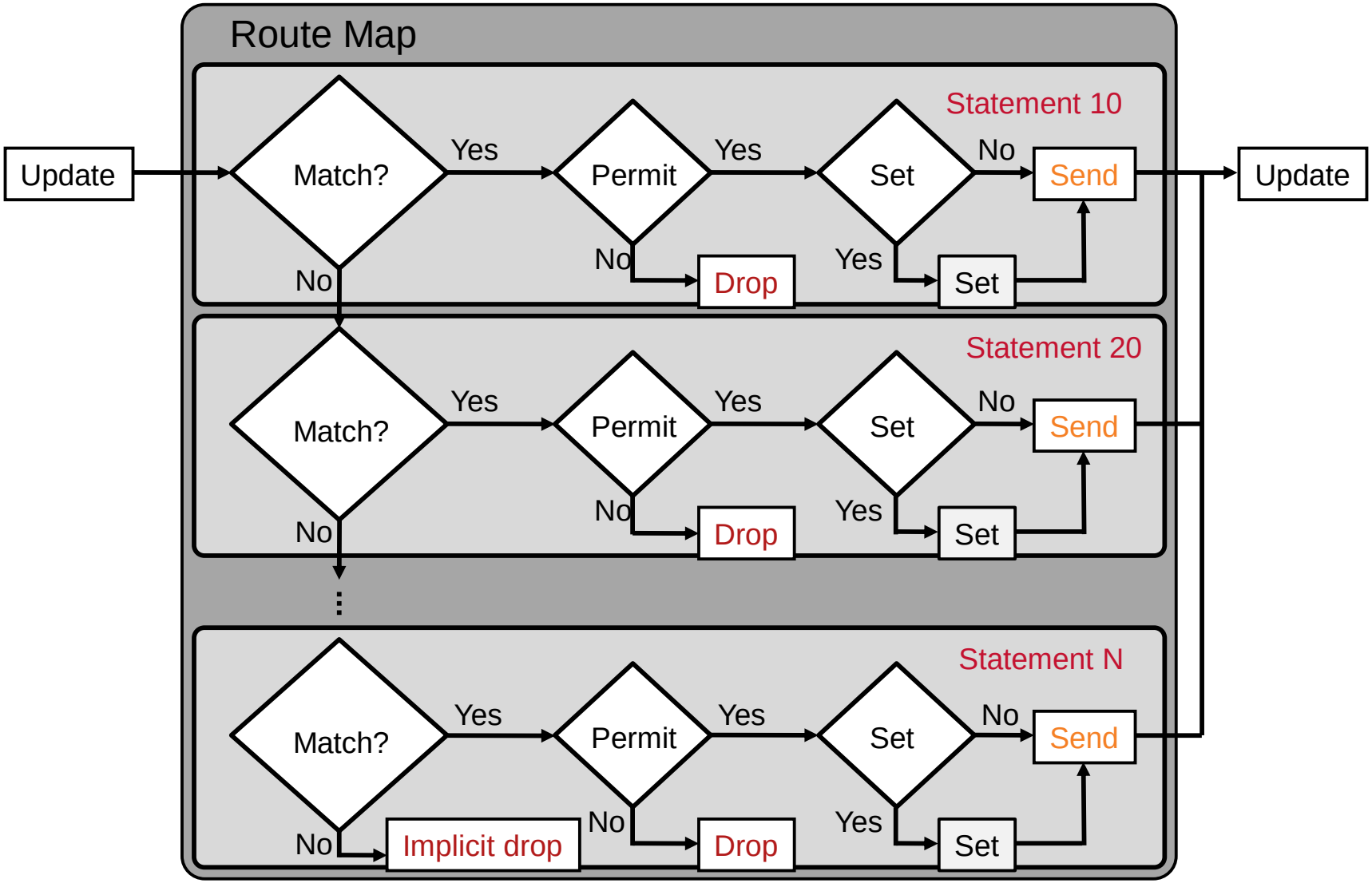
Example: Route Maps

- Preferred paths for specific prefixes
- Backup paths for specific prefixes
- Preferred paths for prefixes based on AS path
- Backup paths for prefixes based on AS path
- Explicit permit at the end

- By default there is a explicit deny all at the end.

```
route-map my_Policy1 permit 10
 match ip address prefix-list PL1
 set local-preference 200
!
route-map my_Policy1 permit 20
 match ip address prefix-list PL2
 set local-preference 50
!
route-map my_Policy1 permit 30
 match as-path APACL1
 set local-preference 200
!
route-map my_Policy1 permit 40
 match as-path APACL2
 set local-preference 50
!
route-map my_Policy1 permit 1000
```

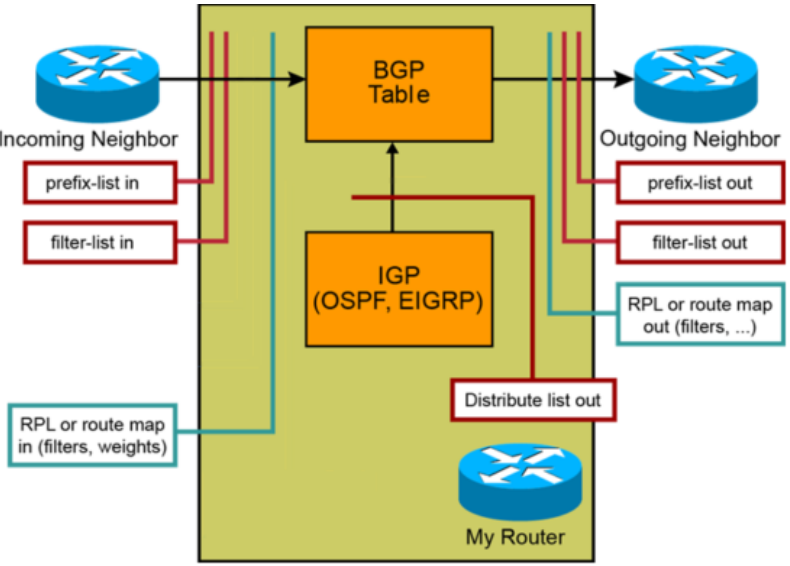
Route Map Processing for Routing Update



BGP Communities Overview

- BGP communities are a means of tagging routes to ensure a consistent filtering or route selection policy.
- The community attribute is a transitive optional attribute. Its value is a 32-bit number (range 0 to 4,294,967,200).
- The standards define several filtering-oriented communities:
  - no-advertise: Do not advertise routes to any peer.
  - no-export: Do not advertise routes to real EBGP peers.
  - local-as: Do not advertise routes to any EBGP peers.
  - internet: Advertise this route to the Internet community.
- A 32-bit community value is split into two parts:
  - High-order 16 bits contain the AS number of the AS that defines the community meaning.
  - Low-order 16 bits have local significance.

Filtering Attachment Points




```
router bgp 1
 neighbor 130.206.212.90 remote-as 2
 neighbor 130.206.212.90 description Upstream1
 neighbor 130.206.212.90 prefix-list BGP_AS1_EXPORT out
 neighbor 130.206.212.90 prefix-list Only_default in
 neighbor 130.206.212.90 route-map RM-Primary in
 neighbor 130.206.212.90 filter-list AS2_Import in
```

```
router bgp 2
 neighbor 130.206.212.89 remote-as 1
 neighbor 130.206.212.89 description Client1
 neighbor 130.206.212.89 prefix-list Only_C1_Nets in
 neighbor 130.206.212.89 route-map RM-LP-Main in
 neighbor 130.206.212.89 filter-list Only_Default out
```